User Modeling in Language Learning with Macaronic Texts

User Modeling in Language Learning with Macaronic Texts

Adithya Renduchintala and **Rebecca Knowles** and **Philipp Koehn** and **Jason Eisner**
Department of Computer Science
Johns Hopkins University
{adi.r,rknowles,phi,eisner}@jhu.edu

Abstract

Foreign language learners can acquire new vocabulary by using cognate and context clues when reading. To measure such incidental comprehension, we devise an experimental framework that involves reading mixed-language "macaronic" sentences. Using data collected via Amazon Mechanical Turk, we train a graphical model to simulate a human subject's comprehension of foreign words, based on cognate clues (edit distance to an English word), context clues (pointwise mutual information), and prior exposure. Our model does a reasonable job at predicting which words a user will be able to understand, which should facilitate the automatic construction of comprehensible text for personalized foreign language education.

1 Introduction

Second language (L2) learning requires the acquisition of vocabulary as well as knowledge of the language's constructions. One of the ways in which learners become familiar with novel vocabulary and constructions is through reading. According to Krashen's Input Hypothesis (Krashen, 1989), learners acquire language through incidental learning, which occurs when learners are exposed to comprehensible input. What constitutes "comprehensible input" for a learner varies as their knowledge of the L2 increases. For example, a student in their first month of German lessons would be hard-pressed to read German novels or even front-page news, but they might understand brief descriptions of daily routines. Comprehensible input need not be completely familiar to the learner; it could include novel vocabulary items or structures (whose meanings they can glean from context). Such input falls in the "zone of proximal development" (Vygotskiĭ, 2012), just outside of the learner's comfort zone. The related concept of "scaffolding" (Wood et al., 1976) consists of providing assistance to the learner at a level that is just sufficient for them to complete their task, which in our case is understanding a sentence.

Automatic selection or construction of comprehensible input—perhaps online and personalized—would be a useful educational technology. However, this requires modeling the student: what can an L2 learner understand in a given context? In this paper, we develop a model and train its parameters on data that we collect.

For the remainder of the paper we focus on native English speakers learning German. Our methodology is a novel solution to the problem of controlling for the learner's German skill level. We use subjects with *zero* previous knowledge of German, but we translate portions of the sentence into English. Thus, we can presume that they *do* already know the English words and *do not* already know the German words (except from seeing them in earlier trials *within* our experiment). We are interested in whether they can jointly infer the meanings of the remaining German words in the sentence, so we ask them to guess.

The resulting stimuli are sentences like "Der Polizist arrested the Bankräuber." Even a reader with no knowledge of German is likely to be able to understand this sentence reasonably well by using cognate and context clues. We refer to this as a *macaronic* sentence; so-called macaronic language is a pastiche of two or more languages (often intended for humorous effect).

Our experimental subjects are required to guess what "Polizist" and "Bankräuber" mean in this sentence. We train a featurized model to predict these guesses jointly within each sentence and thereby predict incidental comprehension on any macaronic sentence. Indeed, we hope our model design will generalize from predicting incidental comprehension on *macaronic* sentences (for our beginner subjects, who need some context words to be in English) to predicting incidental comprehension on *full German* sentences (for more ad-

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 1859–1869,
Berlin, Germany, August 7-12, 2016.

vanced students, who understand some of the context words *as if* they were in English). In addition, we are developing a user interface that uses macaronic sentences directly as a medium of language instruction: our companion paper (Renduchintala et al., 2016) gives an overview of that project.

We briefly review previous work, then describe our data collection setup and the data obtained. Finally, we discuss our model of learner comprehension and validate our model's predictions.

2 Previous Work

Natural language processing (NLP) has long been applied to education, but the majority of this work focuses on evaluation and assessment. Prominent recent examples include Heilman and Madnani (2012), Burstein et al. (2013) and Madnani et al. (2012). Other works fall more along the lines of intelligent and adaptive tutoring systems designed to improve learning outcomes. Most of those are outside of the area of NLP (typically focusing on math or science). An overview of NLP-based work in the education sphere can be found in Litman (2016). There has also been work specific to second language acquisition, such as Özbal et al. (2014), where the focus has been to build a system to help learners retain new vocabulary. However, much of the existing work on incidental learning is found in the education and cognitive science literature rather than NLP.

Our work is related to Labutov and Lipson (2014), which also tries to leverage incidental learning using mixed L1 and L2 language. Where their work uses surprisal to choose contexts in which to insert L2 vocabulary, we consider both context features and other factors such as cognate features (described in detail in 4.1). We collect data that gives direct evidence of the user's understanding of words (by asking them to provide English guesses) rather than indirectly (via questions about sentence validity, which runs the risk of overestimating their knowledge of a word, if, for instance, they've only learned whether it is animate or inanimate rather than the exact meaning). Furthermore, we are not only interested in whether a mixed L1 and L2 sentence is comprehensible; we are also interested in determining a distribution over the learner's belief state for each word in the sentence. We do this in an engaging, game-like setting, which provides the user with hints when the task is too difficult for them to complete.

3 Data Collection Setup

Our method of scaffolding is to replace certain foreign words and phrases with their English translations, yielding a macaronic sentence.[1] Simply presenting these to a learner would not give us feedback on the learner's belief state for each foreign word. Even assessing the learner's reading comprehension would give only weak indirect information about what was understood. Thus, we collect data where a learner explicitly guesses a foreign word's translation when seen in the macaronic context. These guesses are then treated as supervised labels to train our user model.

We used Amazon Mechanical Turk (MTurk) to collect data. Users qualified for tasks by completing a short quiz and survey about their language knowledge. Only users whose results indicated no knowledge of German and self-identified as native speakers of English were allowed to complete tasks. With German as the foreign language, we generated content by crawling a simplified-German news website, `nachrichtenleicht.de`. We chose simplified German in order to minimize translation errors and to make the task more suitable for novice learners. We translated each German sentence using the Moses Statistical Machine Translation (SMT) toolkit (Koehn et al., 2007). The SMT system was trained on the German-English Commoncrawl parallel text used in WMT 2015 (Bojar et al., 2015).

We used 200 German sentences, presenting each to 10 different users. In MTurk jargon, this yielded 2000 Human Intelligence Tasks (HITs). Each HIT required its user to participate in several rounds of guessing as the English translation was incrementally revealed. A user was paid US$0.12 per HIT, with a bonus of US$6 to any user who accumulated more than 2000 total points.

Our HIT user interface is shown in the video at `https://youtu.be/9PczEcnr4F8`.

3.1 HITs and Submissions

For each HIT, the user first sees a German sentence[2] (Figure 1). A text box is presented below each German word in the sentence, for the user

[1] Although the language distinction is indicated by italics and color, users were left to figure this out on their own.

[2] Except that we first "translate" any German words that have identical spelling in English (case-insensitive). This includes most proper names, numerals, and punctuation marks. Such translated words are displayed in English style (blue italics), and the user is not asked to guess their meanings.

Figure 1: After a user submits a set of guesses (top), the interface marks the correct guesses in green and also reveals a set of translation clues (bottom). The user now has the opportunity to guess again for the remaining German words.

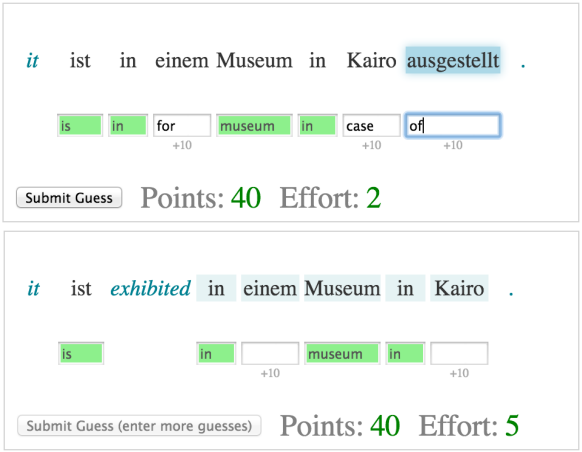


Figure 2: In this case, after the user submits a set of guesses (top), two clues are revealed (bottom): `ausgestellt` is moved into English order and then translated.

to type in their "best guess" of what each German word means. The user must fill in at least half of the text boxes before submitting this set of guesses. The resulting *submission*—i.e., the macaronic sentence together with the set of guesses—is logged in a database as a single training example, and the system displays feedback to the user about which guesses were correct.

After each submission, new *clues* are revealed (providing increased scaffolding) and the user is asked to guess again. The process continues, yielding multiple submissions, until all German words in the sentence have been translated. At this point, the entire HIT is considered completed and the user moves to a new HIT (i.e., a new sentence).

From our 2000 HITs, we obtained 9392 submissions (4.7 per HIT) from 79 distinct MTurk users.

3.2 Clues

Each update provides new clues to help the user make further guesses. There are 2 kinds of clues:

Translation Clue (Figure 1): A set of words that were originally in German are replaced with their English translations. The text boxes below these words disappear, since it is no longer necessary to guess them.

Reordering Clue (Figure 2): A German substring is moved into a more English-like position. The reordering positions are calculated using the word and phrase alignments obtained from Moses.

Each time the user submits a set of guesses, we reveal a sequence of $n = \max(1, \text{round}(N/3))$ clues, where N is the number of German words remaining in the sentence. For each clue, we sample a token that is currently in German. If the token is part of a movable phrase, we move that phrase; otherwise we translate the minimal phrase containing that token. These moves correspond exactly to clues that a user could request by clicking on the token in the macaronic reading interface of Renduchintala et al. (2016)—see that paper for details of how moves are constructed and animated. In our present experiments, the system is in control instead, and grants clues by "randomly clicking" on n tokens.

The system's probability of sampling a given token is proportional to its unigram type probability in the WMT corpus. Thus, rarer words tend to remain in German for longer, allowing the Turker to attempt more guesses for these difficult words.

3.3 Feedback

When a user submits a set of guesses, the system responds with feedback. Each guess is visibly "marked" in left-to-right order, momentarily shaded with green (for correct), yellow (for close) or red (for incorrect). Depending on whether a guess is correct, close, or wrong, users are awarded points as discussed below. Yellow and red shading then fades, to signal to the user that they may try entering a new guess. Correct guesses remain on the screen for the entire task.

3.4 Points

Adding points to the process (Figures 1–2) adds a game-like quality and lets us incentivize users by paying them for good performance (see section 3). We award 10 points for each exactly correct guess (case-insensitive). We give additional "effort points" for a guess that is close to the cor-

rect translation, as measured by cosine similarity in vector space. (We used pre-trained GLoVe word vectors (Pennington et al., 2014); when the guess or correct translation has multiple words, we take the average of the word vectors.) We deduct effort points for guesses that are careless or very poor. Our rubric for effort points is as follows:

$$e_p = \begin{cases} -1, & \text{if } \hat{e} \text{ is repeated or nonsense (red)} \\ -1, & \text{if } \mathrm{sim}(\hat{e}, e^*) < 0 \text{ (red)} \\ 0, & \text{if } 0 \leq \mathrm{sim}(\hat{e}, e^*) < 0.4 \text{ (red)} \\ 0, & \text{if } \hat{e} \text{ is blank} \\ 10 \times \mathrm{sim}(\hat{e}, e^*) & \text{otherwise (yellow)} \end{cases}$$

Here $\mathrm{sim}(\hat{e}, e^*)$ is cosine similarity between the vector embeddings of the user's guess $\hat{e}$ and our reference translation e^*. A "nonsense" guess contains a word that does not appear in the sentence bitext nor in the 20,000 most frequent word types in the GLoVe training corpus. A "repeated" guess is an incorrect guess that appears more than once in the set of guesses being submitted.

In some cases, $\hat{e}$ or e^* may itself consist of multiple words. In this case, our points and feedback are based on the best match between any word of $\hat{e}$ and any word of e^*. In alignments where multiple German words translate as a single phrase,[3] we take the phrasal translation to be the correct answer e^* for *each* of the German words.

3.5 Normalization

After collecting the data, we normalized the user guesses for further analysis. All guesses were lowercased. Multi-word guesses were crudely replaced by the longest word in the guess (breaking ties in favor of the earliest word).

The guesses included many spelling errors as well as some nonsense strings and direct copies of the input. We defined the *dictionary* to be the 100,000 most frequent word types (lowercased) from the WMT English data. If a user's guess $\hat{e}$ does not match e^* and is not in the dictionary, we replace it with

- the special symbol <COPY>, if $\hat{e}$ appears to be a copy of the German source word f (meaning that its Levenshtein distance from f is $< 0.2 \cdot \max(|\hat{e}|, |f|)$);
- else, the closest word in the dictionary[4] as measured by Levenshtein distance (breaking ties alphabetically), provided the dictionary has a word at distance ≤ 2;
- else <BLANK>, as if the user had not guessed.

4 User Model

In each submission, the user jointly guesses several English words, given spelling and context clues. One way that a *machine* could perform this task is via probabilistic inference in a factor graph—and we take this as our model of how the *human user* solves the problem.

The user observes a German sentence $\mathbf{f} = [f_1, f_2, \ldots, f_i, \ldots f_n]$. The translation of each word token f_i is E_i, which is from the user's point of view a random variable. Let Obs denote the set of indices i for which the user also observes that $E_i = e_i^*$, the aligned reference translation, because e_i^* has already been guessed correctly (green feedback) or shown as a clue. Thus, the user's posterior distribution over $\mathbf{E}$ is $P_{\boldsymbol{\theta}}(\mathbf{E} = \mathbf{e} \mid \mathbf{E}_{\text{Obs}} = \mathbf{e}^*_{\text{Obs}}, \mathbf{f}, \text{history})$, where "history" denotes the user's history of past interactions.

We assume that a user's submission $\hat{\mathbf{e}}$ is derived from this posterior distribution simply as a random sample. We try to fit the parameter vector $\boldsymbol{\theta}$ to maximize the log-probability of the submission. Note that our model is trained on the user guesses $\hat{\mathbf{e}}$, not the reference translations $\mathbf{e}^*$. That is, we seek parameters $\boldsymbol{\theta}$ that would explain why all users made their guesses.

Although we fit a single $\boldsymbol{\theta}$, this does not mean that we treat users as interchangeable (since $\boldsymbol{\theta}$ can include user-specific parameters) or unvarying (since our model conditions users' behavior on their history, which can capture some learning).

4.1 Factor Graph

We model the posterior distribution as a conditional random field (Figure 3) in which the value of E_i depends on the form of f_i as well as on the meanings e_j (which may be either observed or jointly guessed) of the context words at $j \neq i$:

$$P_{\boldsymbol{\theta}}(\mathbf{E} = \mathbf{e} \mid \mathbf{E}_{\text{Obs}} = \mathbf{e}^*_{\text{Obs}}, \mathbf{f}, \text{history}) \quad (1)$$
$$\propto \prod_{i \notin \text{Obs}} (\psi^{\text{ef}}(e_i, f_i) \cdot \prod_{j \neq i} \psi^{\text{ee}}(e_i, e_j, i-j))$$

We will define the factors ψ (the *potential functions*) in such a way that they do not "know German" but only have access to information that is available to an naive English speaker. In brief, the

[3] Our German-English alignments are constructed as in Renduchintala et al. (2016).

[4] Considering only words returned by the Pyenchant 'suggest' function (http://pythonhosted.org/pyenchant/).

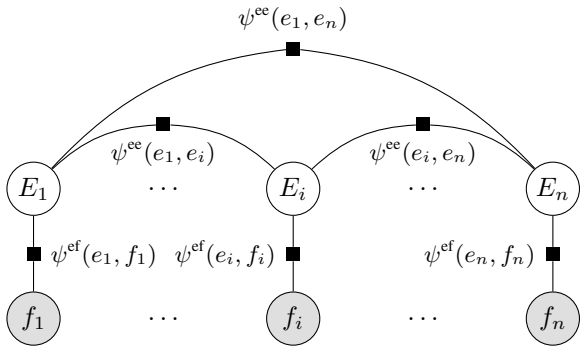


Figure 3: Model for user understanding of L2 words in sentential context. This figure shows an inference problem in which all the observed words in the sentence are in German (that is, Obs $= \emptyset$). As the user observes translations via clues or correctly-marked guesses, some of the E_i become shaded.

factor $\psi^{\text{ef}}(e_i, f_i)$ considers whether the hypothesized English word e_i "looks like" the observed German word f_i, and whether the user has previously observed during data collection that e_i is a correct or incorrect translation of f_i. Meanwhile, the factor $\psi^{\text{ee}}(e_i, e_j)$ considers whether e_i is commonly seen in the context of e_j in English text. For example, the user will elevate the probability that $E_i =$ `cake` if they are fairly certain that E_j is a related word like `eat` or `chocolate`.

The potential functions ψ are parameterized by $\boldsymbol{\theta}$, a vector of feature weights. For convenience, we define the features in such a way that we expect their weights to be positive. We rely on just 6 features at present (see section 6 for future work), although each is complex and real-valued. Thus, the weights $\boldsymbol{\theta}$ control the relative influence of these 6 different types of information on a user's guess. Our features broadly fall under the following categories: *Cognate*, *History*, and *Context*. We precomputed cognate and context features, while history features are computed on-the-fly for each training instance. All features are case-insensitive.

4.1.1 Cognate and History Features

For each German token f_i, the ψ^{ef} factor can score each possible guess e_i of its translation:

$$\psi^{\text{ef}}(e_i, f_i) = \exp(\boldsymbol{\theta}^{\text{ef}} \cdot \boldsymbol{\phi}^{\text{ef}}(e_i, f_i)) \quad (2)$$

The feature function $\boldsymbol{\phi}^{\text{ef}}$ returns a vector of 4 real numbers:

- *Orthographic Similarity*: The normalized Levenshtein distance between the 2 strings.

$$\phi^{\text{ef}}_{\text{orth}}(e_i, f_i) = 1 - \frac{\text{lev}(e_i, f_i)}{\max(|e_i|, |f_i|)} \quad (3)$$

  The weight on this feature encodes how much users pay attention to spelling.

- *Pronunciation Similarity*: This feature is similar to the previous one, except that it calculates the normalized distance between the *pronunciations* of the two words:

$$\phi^{\text{ef}}_{\text{pron}}(e_i, f_i) = \phi^{\text{ef}}_{\text{orth}}(\text{prn}(e_i), \text{prn}(f_i)) \quad (4)$$

  where the function $\text{prn}(x)$ maps a string x to its pronunciation. We obtained pronunciations for all words in the English and German vocabularies using the CMU pronunciation dictionary tool (Weide, 1998). Note that we use English pronunciation rules even for German words. This is because we are modeling a naive learner who may, in the absence of intuition about German pronunciation rules, apply English pronunciation rules to German.

- *Positive History Feature*: If a user has been rewarded in a previous HIT for guessing e_i as a translation of f_i, then they should be more likely to guess it again. We define $\boldsymbol{\phi}^{\text{ef}}_{\text{hist+}}(e_i, f_i)$ to be 1 in this case and 0 otherwise. The weight on this feature encodes whether users learn from positive feedback.

- *Negative History Feature*: If a user has already incorrectly guessed e_i as a translation of f_i in a previous submission during this HIT, then they should be less likely to guess it again. We define $\boldsymbol{\phi}^{\text{ef}}_{\text{hist-}}(e_i, f_i)$ to be -1 in this case and 0 otherwise. The weight on this feature encodes whether users remember negative feedback.[5]

4.1.2 Context Features

In the same way, the ψ^{ef} factor can score the compatibility of a guess e_i with a context word e_j, which may itself be a guess, or may be observed:

$$\psi^{\text{ee}}_{ij}(e_i, e_j) = \exp(\boldsymbol{\theta}^{\text{ee}} \cdot \boldsymbol{\phi}^{\text{ee}}(e_i, e_j, i-j)) \quad (5)$$

$\boldsymbol{\phi}^{\text{ee}}$ returns a vector of 2 real numbers:

$$\boldsymbol{\phi}^{\text{ee}}_{\text{pmi}}(e_i, e_j) = \begin{cases} \text{PMI}(e_i, e_j) \text{ if } |i-j| > 1 \\ 0 \text{ otherwise} \end{cases} \quad (6)$$

$$\boldsymbol{\phi}^{\text{ee}}_{\text{pmi1}}(e_i, e_j) = \begin{cases} \text{PMI}_1(e_i, e_j) \text{ if } |i-j| = 1 \\ 0 \text{ otherwise} \end{cases} \quad (7)$$

[5] At least in short-term memory—this feature currently omits to consider any negative feedback from previous HITs.

where the pointwise mutual information $\mathrm{PMI}(x, y)$ measures the degree to which the English words x, y tend to occur in the same English sentence, and $\mathrm{PMI}_1(x, y)$ measures how often they tend to occur in adjacent positions. These measurements are estimated from the English side of the WMT corpus, with smoothing performed as in Knowles et al. (2016).

For example, if $f_i =$ `Suppe`, the user's guess of E_i should be influenced by $f_j =$ `Brot` appearing in the same sentence, if the user suspects or observes that its translation is $E_j =$ `bread`. The PMI feature knows that `soup` and `bread` tend to appear in the same English sentences, whereas PMI_1 knows that they tend not to appear in the bigram `soup bread` or `bread soup`.

4.1.3 User-Specific Features

Apart from the basic 6-feature model, we also trained a version that includes user-specific copies of each feature (similar to the domain adaptation technique of Daumé III (2007)). For example, $\phi^{\text{ef}}_{\text{orth},32}(e_i, f_i)$ is defined to equal $\phi^{\text{ef}}_{\text{orth}}(e_i, f_i)$ for submissions by user 32, and defined to be 0 for submissions by other users.

Thus, with 79 users in our dataset, we learned 6×80 feature weights: a local weight vector for each user and a global vector of "backoff" weights. The global weight $\theta^{\text{ef}}_{\text{orth}}$ is large if users in general reward orthographic similarity, while $\theta^{\text{ef}}_{\text{orth},32}$ (which may be positive or negative) captures the degree to which user 32 rewards it more or less than is typical. The user-specific features are intended to capture individual differences in incidental comprehension.

4.2 Inference

According to our model, the probability that the user guesses $E_i = \hat{e}_i$ is given by a marginal probability from the CRF. Computing these marginals is a combinatorial optimization problem that involves reasoning jointly about the possible values of each E_i ($i \notin$ Obs), which range over the English vocabulary V^{e}.

We employ loopy belief propagation (Murphy et al., 1999) to obtain approximate marginals over the variables $\mathbf{E}$. A *tree-based* schedule for message passing was used (Dreyer and Eisner, 2009, footnote 22). We run 3 iterations with a new random root for each iteration.

We define the vocabulary V^{e} to consist of all reference translations e^*_i and normalized user guesses $\hat{e}_i$ from our entire dataset (see section 3.5), about 5K types altogether including `<BLANK>` and `<COPY>`. We define the cognate features to treat `<BLANK>` as the empty string and to treat `<COPY>` as f_i. We define the PMI of these special symbols with any e to be the mean PMI with e of all dictionary words, so that they are essentially uninformative.

4.3 Parameter Estimation

We learn our parameter vector $\boldsymbol{\theta}$ to approximately maximize the regularized log-likelihood of the users' guesses:

$$\left(\sum \log P_{\boldsymbol{\theta}}(\mathbf{E} = \hat{\mathbf{e}} \mid \mathbf{E}_{\text{Obs}} = \mathbf{e}^*_{\text{Obs}}, \mathbf{f}, \text{history})\right) - \lambda ||\boldsymbol{\theta}||^2 \quad (8)$$

where the summation is over all submissions in our dataset. The gradient of each summand reduces to a difference between observed and expected values of the feature vector $\boldsymbol{\phi} = (\boldsymbol{\phi}^{\text{ef}}, \boldsymbol{\phi}^{\text{ee}})$, summed over all factors in (1). The observed features are computed directly by setting $\mathbf{E} = \hat{\mathbf{e}}$. The expected features (which arise from the log of the normalization constant of (1)) are computed approximately by loopy belief propagation.

We trained $\boldsymbol{\theta}$ using stochastic gradient descent (SGD),[6] with a learning rate of 0.1 and regularization parameter of 0.2. The regularization parameter was tuned on our development set.

5 Experimental Results

We divided our data randomly into 5550 training instances, 1903 development instances, and 1939 test instances. Each instance was a single submission from one user, consisting of a batch of "simultaneous" guesses on a macaronic sentence.

We noted qualitatively that when a large number of English words have been revealed, particularly content words, the users tend to make better guesses. Conversely, when most context is German, we unsuprisingly see the user leave many guesses blank and make other guesses based on string similarity triggers. Such submissions are difficult to predict as different users will come up with a wide variety of guesses; our model therefore resorts to predicting similar-sounding words. For detailed examples of this see Appendix A.

[6] To speed up training, SGD was parallelized using Recht et al.'s (2011) Hogwild! algorithm. We trained for 8 epochs.

Model	Recall at k (dev)			Recall at k (test)		
	1	25	50	1	25	50
Basic	15.24	34.26	38.08	16.14	35.56	40.30
User-Adapted	15.33	34.40	38.67	16.45	35.71	40.57

Table 1: Percentage of foreign words for which the user's actual guess appears in our top-k list of predictions, for models with and without user-specific features ($k \in \{1, 25, 50\}$).

For each foreign word f_i in a submission with $i \notin$ Obs, our inference method (section 4.2) predicts a marginal probability distribution over a user's guesses $\hat{e}_i$. Table 1 shows our ability to predict user guesses.[7] Recall that this task is essentially a structured prediction task that does joint 4919-way classification of each German word. Roughly 1/3 of the time, our model's top 25 words include the user's exact guess.

However, the recall reported in Table 1 is too stringent for our educational application. We could give the model partial credit for predicting a synonym of the learner's guess $\hat{e}$. More precisely, we would like to give the model partial credit for predicting when the learner will make a poor guess of the truth e^*—even if the model does not predict the user's specific incorrect guess $\hat{e}$.

To get at this question, we use English word embeddings (as in section 3.4) as a proxy for the semantics and morphology of the words. We measure the *actual quality* of the learner's guess $\hat{e}$ as its cosine similarity to the truth, $\mathrm{sim}(\hat{e}, e^*)$. While quality of 1 is an exact match, and quality scores > 0.75 are consistently good matches, we found quality of ≈ 0.6 also reasonable. Pairs such as (`mosque`, `islamic`) and (`politics`, `government`) are examples from the collected data with quality ≈ 0.6. As quality becomes < 0.4, however, the relationship becomes tenuous, e.g., (`refugee`, `soil`).

Similarly, we measure the *predicted quality* as $\mathrm{sim}(e, e^*)$, where e is the model's 1-best prediction of the user's guess. Figure 4 plots predicted vs. actual quality (each point represents one of the learner's guesses on development data), obtaining a correlation of 0.38, which we call the "quality correlation" or QC. A clear diagonal band can be seen, corresponding to the instances where the model exactly predicts the user's guess. The cloud around the diagonal is formed by instances where the model's prediction was not identical to the user's guess but had similar quality.

[7] Throughout this section, we ignore the 5.2% of tokens on which the user did not guess (i.e., the guess was `<BLANK>` after the normalization of section 3.5). Our present model simply treats `<BLANK>` as an ordinary and very bland word (section 4.2), rather than truly attempting to predict when the user will not guess. Indeed, the model's posterior probability of `<BLANK>` in these cases is a paltry 0.0000267 on average (versus 0.0000106 when the user does guess). See section 6.

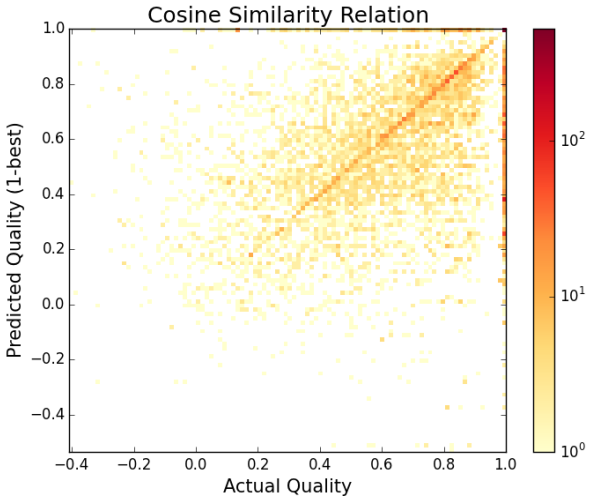


Figure 4: Actual quality $\mathrm{sim}(\hat{e}, e^*)$ of the learner's guess $\hat{e}$ on development data, versus predicted quality $\mathrm{sim}(e, e^*)$ where e is the basic model's 1-best prediction.

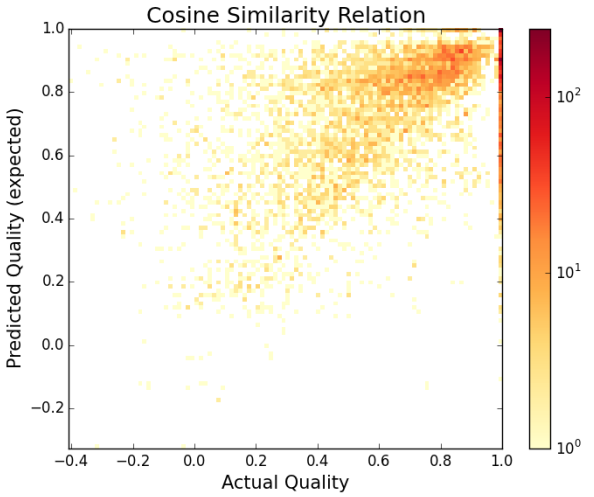


Figure 5: Actual quality $\mathrm{sim}(\hat{e}, e^*)$ of the learner's guess $\hat{e}$ on development data, versus the expectation of the predicted quality $\mathrm{sim}(e, e^*)$ where e is distributed according to the basic model's posterior.

We also consider the *expected* predicted quality, averaging over the model's predictions e of $\hat{e}$ (for all $e \in V^e$) in proportion to the probabilities that it assigns them. This allows the model to more smoothly assess whether the learner is likely to make a high-quality guess. Figure 5 shows this version, where the points tend to shift upward and the quality correlation (QC) rises to 0.53.

All QC values are given in Table 2. We used expected QC on the development set as the criterion for selecting the regularization coefficient λ and as the early stopping criterion during training.

Model	Dev		Test	
	Exp	1-Best	Exp	1-Best
Basic	0.525	0.379	0.543	0.411
User-Adapted	0.527	0.427	0.544	0.439

Table 2: Quality correlations: basic and user-adapted models.

Feature Removed	QC	
	Expected	1-Best
None	0.522	0.425
Cognate	0.516	0.366*
Context	0.510	0.366*
History	0.499*	0.259*

Table 3: Impact on quality correlation (QC) of removing features from the model. Ablated QC values marked with asterisk* differ significantly from the full-model QC values in the first row ($p < 0.05$, using the test of Preacher (2002)).

5.1 Feature Ablation

To test the usefulness of different features, we trained our model with various feature categories disabled. To speed up experimentation, we sampled 1000 instances from the training set and trained our model on those. The resulting QC values on dev data are shown in Table 3. We see that removing history-based features has the most significant impact on model performance: both QC measures drop relative to the full model. For cognate and context features, we see no significant impact on the expected QC, but a significant drop in the 1-best QC, especially for context features.

5.2 Analysis of User Adaptation

Table 2 shows that the user-specific features significantly improve the *1-best* QC of our model, although the much smaller improvement in *expected* QC is insignificant.

User adaptation allows us to discern different styles of incidental comprehension. A user-adapted model makes fine-grained predictions that could help to construct better macaronic sentences for a given user. Each user who completed at least 10 HITs has their user-specific weight vector shown as a row in Figure 6. Recall that the user-specific weights are not used in isolation, but are *added* to backoff weights shared by all users.

These user-specific weight vectors cluster into four groups. Furthermore, the average points per HIT differ by cluster (significantly between each cluster pair), reflecting the success of different strategies.[8] Users in group (a) employ a generalist

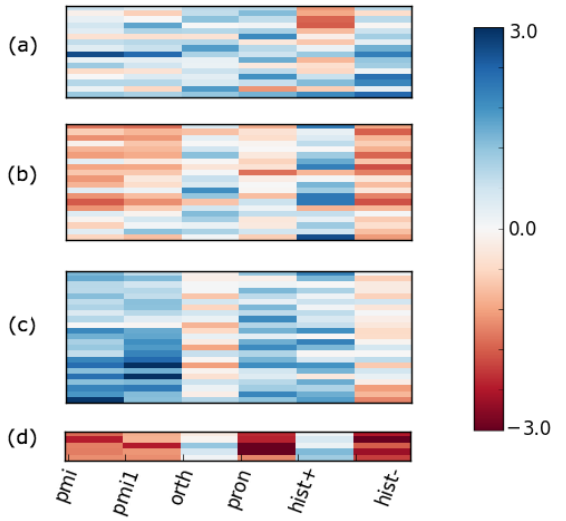


Figure 6: The user-specific weight vectors, clustered into groups. Average points per HIT for the HITs completed by each group: (a) 45, (b) 48, (c) 50 and (d) 42.

strategy for incidental comprehension. They pay typical or greater-than-typical attention to all features of the current HIT, but many of them have diminished memory for vocabulary learned during past HITs (the hist+ feature). Users in group (b) seem to use the opposite strategy, deriving their success from retaining common vocabulary across HITs (hist+) and falling back on orthography for new words. Group (c) users, who earned the most points per HIT, appear to make heavy use of context and pronunciation features *together* with hist+. We also see that pronunciation similarity seems to be a stronger feature for group (c) users, in contrast to the more superficial orthographic similarity. Group (d), which earned the fewest points per HIT, appears to be an "extreme" version of group (b): these users pay unusually little attention to any model features other than orthographic similarity and hist+. (More precisely, the model finds group (d)'s guesses harder to predict on the basis of the available features, and so gives a more uniform distribution over V^e.)

6 Future Improvements to the Model

Our model's feature set (section 4.1) could clearly be refined and extended. Indeed, in a separate paper (Knowles et al., 2016), we use a more tightly controlled experimental design to explore some simple feature variants. A cheap way to vet features would be to test whether they help on the task of modeling reference translations, which are

[8] Recall that in our data collection process, we award points for each HIT (section 3.4). While the points were designed more as a reward than as an evaluation of learner success, a higher score does reflect more guesses that were correct or close, while a lower score indicates that some words were never guessed before the system revealed them as clues.

more plentiful and less noisy than the user guesses.

For *Cognate* features, there exist many other good string similarity metrics (including trainable ones). We could also include $\boldsymbol{\phi}^{\text{ef}}$ features that consider whether e_i's part of speech, frequency, and length are plausible given f_i's burstiness, observed frequency, and length. (E.g., only short common words are plausibly translated as determiners.)

For *Context* features, we could design versions that are more sensitive to the position and status of the context word j. We speculate that the actual influence of e_j on a user's guess e_i is stronger when e_j is observed rather than itself guessed; when there are fewer intervening tokens (and particularly fewer observed ones); and when $j < i$. Orthogonally, $\boldsymbol{\phi}^{\text{ef}}(e_i, e_j)$ could go beyond PMI and windowed PMI to also consider cosine similarity, as well as variants of these metrics that are thresholded or nonlinearly transformed. Finally, we do not have to treat the context positions j as independent multiplicative influences as in equation (1) (cf. Naive Bayes): we could instead use a topic model or some form of language model to determine a conditional probability distribution over E_i given all other words in the context.

An obvious gap in our current feature set is that we have no $\boldsymbol{\phi}^{\text{e}}$ features to capture that some words $e_i \in V^{\text{e}}$ are more likely guesses *a priori*. By defining several versions of this feature, based on frequencies in corpora of different reading levels, we could learn user-specific weights modeling which users are unlikely to think of an obscure word. We should also include features that fire specifically on the reference translation e_i^* and the special symbols <BLANK> and <COPY>, as each is much more likely than the other features would suggest.

For *History* features, we could consider *negative* feedback from other HITs (not just the current HIT), as well as *positive* information provided by revealed clues (not just confirmed guesses). We could also devise non-binary versions in which more recent or more frequent feedback on a word has a stronger effect. More ambitiously, we could model generalization: after being shown that `Kind` means `child`, a learner might increase the probability that the similar word `Kinder` means `child` or something related (`children`, `childish`, ...), whether because of superficial orthographic similarity or a deeper understanding of the morphology. Similarly, a learner might gradually acquire a model of typical spelling changes in English-German cognate pairs.

A more significant extension would be to model a user's learning process. Instead of representing each user by a small vector of user-specific weights, we could recognize that the user's guessing strategy and knowledge can change over time.

A serious deficiency in our current model (not to mention our evaluation metrics!) is that we treat <BLANK> like any other word. A more attractive approach would be to learn a stochastic link from the posterior distribution to the user's guess or non-guess, instead of assuming that the user simply samples the guess from the posterior. As a simple example, we might say the user guesses $e \in V^{\text{e}}$ with probability $p(e)^\beta$—where $p(e)$ is the posterior probability and $\beta > 1$ is a learned parameter—with the remaining probability assigned to <BLANK>. This says that the user tends to avoid guessing except when there are relatively high-probability words to guess.

7 Conclusion

We have presented a methodology for collecting data and training a model to estimate a foreign language learner's understanding of L2 vocabulary in partially understood contexts. Both are novel contributions to the study of L2 acquisition.

Our current model is arguably crude, with only 6 features, yet it can already often do a reasonable job of predicting what a user might guess and whether the user's guess will be roughly correct. This opens the door to a number of future directions with applications to language acquisition using personalized content and learners' knowledge.

We plan a deeper investigation into how learners detect and combine cues for incidental comprehension. We also leave as future work the integration of this model into an adaptive system that tracks learner understanding and creates scaffolded content that falls in their zone of proximal development, keeping them engaged while stretching their understanding.

Acknowledgments

This work was supported by a seed grant from the Science of Learning Institute at Johns Hopkins University, and also by a National Science Foundation Graduate Research Fellowship (Grant No. DGE-1232825) to the second author. We thank Chadia Abras, Adam Teichert, and Sanjeev Khudanpur for helpful discussions and suggestions.

References

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Barry Haddow, Matthias Huck, Chris Hokamp, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Carolina Scarton, Lucia Specia, and Marco Turchi. Findings of the 2015 Workshop on Statistical Machine Translation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 1–46, 2015.

Jill Burstein, Joel Tetreault, and Nitin Madnani. The e-rater automated essay scoring system. In Mark D. Shermis, editor, *Handbook of Automated Essay Evaluation: Current Applications and New Directions*, pages 55–67. Routledge, 2013.

Hal Daumé III. Frustratingly easy domain adaptation. In *Proceedings of ACL*, pages 256–263, June 2007.

Markus Dreyer and Jason Eisner. Graphical models over multiple strings. In *Proceedings of EMNLP*, pages 101–110, Singapore, August 2009.

Michael Heilman and Nitin Madnani. ETS: Discriminative edit models for paraphrase scoring. In *Joint Proceedings of *SEM and SemEval*, pages 529–535, June 2012.

Rebecca Knowles, Adithya Renduchintala, Philipp Koehn, and Jason Eisner. Analyzing learner understanding of novel L2 vocabulary. 2016. To appear.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL: Short Papers*, pages 177–180, 2007.

Stephen Krashen. We acquire vocabulary and spelling by reading: Additional evidence for the input hypothesis. *The Modern Language Journal*, 73(4):440–464, 1989.

Igor Labutov and Hod Lipson. Generating code-switched text for lexical learning. In *Proceedings of ACL*, pages 562–571, 2014.

Diane Litman. Natural language processing for enhancing teaching and learning. In *Proceedings of AAAI*, 2016.

Nitin Madnani, Michael Heilman, Joel Tetreault, and Martin Chodorow. Identifying high-level organizational elements in argumentative discourse. In *Proceedings of NAACL-HLT*, pages 20–28, 2012.

Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of UAI*, pages 467–475, 1999.

Gözde Özbal, Daniele Pighin, and Carlo Strapparava. Automation and evaluation of the keyword method for second language learning. In *Proceedings of ACL (Volume 2: Short Papers)*, pages 352–357, 2014.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GLoVe: Global vectors for word representation. In *Proceedings of EMNLP*, volume 14, pages 1532–1543, 2014.

K. J. Preacher. Calculation for the test of the difference between two independent correlation coefficients [computer software], May 2002.

Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 693–701, 2011.

Adithya Renduchintala, Rebecca Knowles, Philipp Koehn, and Jason Eisner. Creating interactive macaronic interfaces for language learning. In *Proceedings of ACL (System Demonstrations)*, 2016.

Lev Vygotskiĭ. *Thought and Language (Revised and Expanded Edition)*. MIT Press, 2012.

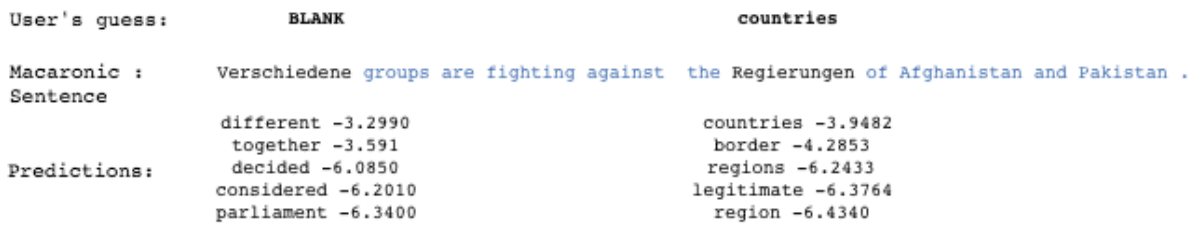
R. Weide. The CMU pronunciation dictionary, release 0.6, 1998.

David Wood, Jerome S. Bruner, and Gail Ross. The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry*, 17(2): 89–100, 1976.

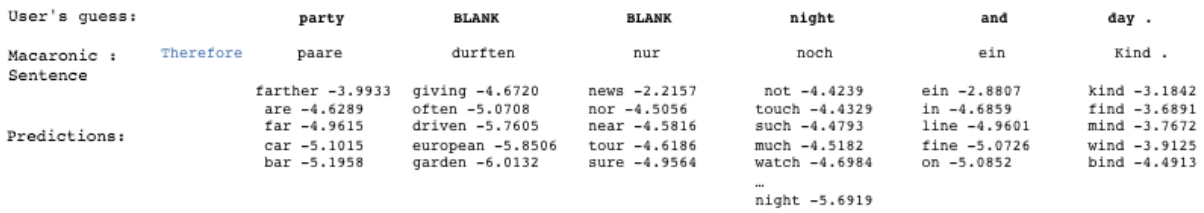
Appendices

A Example of Learner Guesses vs. Model Predictions

To give a sense of the problem difficulty, we have hand-picked and presented two training examples (submissions) along with the predictions of our basic model and their log-probabilities. In Figure 7a a large portion of the sentence has been revealed to the user in English (blue text) only 2 words are in German. The text in bold font is the user's guess. Our model expected both words to be guessed; the predictions are listed below the German words `Verschiedene` and `Regierungen`. The reference translation for the 2 words are `Various` and `governments`. In Figure 7b we see a much harder context where only one word is shown in English and this word is not particularly helpful as a contextual anchor.



(a)



(b)

Figure 7: Two examples of the system's predictions of what the user will guess on a single submission, contrasted with the user's actual guess. (The user's previous submissions on the same task instance are not shown.) In 7a, the model correctly expects that the substantial context will inform the user's guess. In 7b, the model predicts that the user will fall back on string similarity—although we can see that the user's actual guess of `and day` was likely informed by their guess of `night`, an influence that our CRF did consider. The numbers shown are log-probabilities. Both examples show the sentences in a macaronic state (after some reordering or translation has occurred). For example, the original text of the German sentence in 7b reads `Deshalb durften die Paare nur noch ein Kind bekommen .` The macaronic version has undergone some reordering, and has also erroneously dropped the verb due to an incorrect alignment.